Manuel d'installation

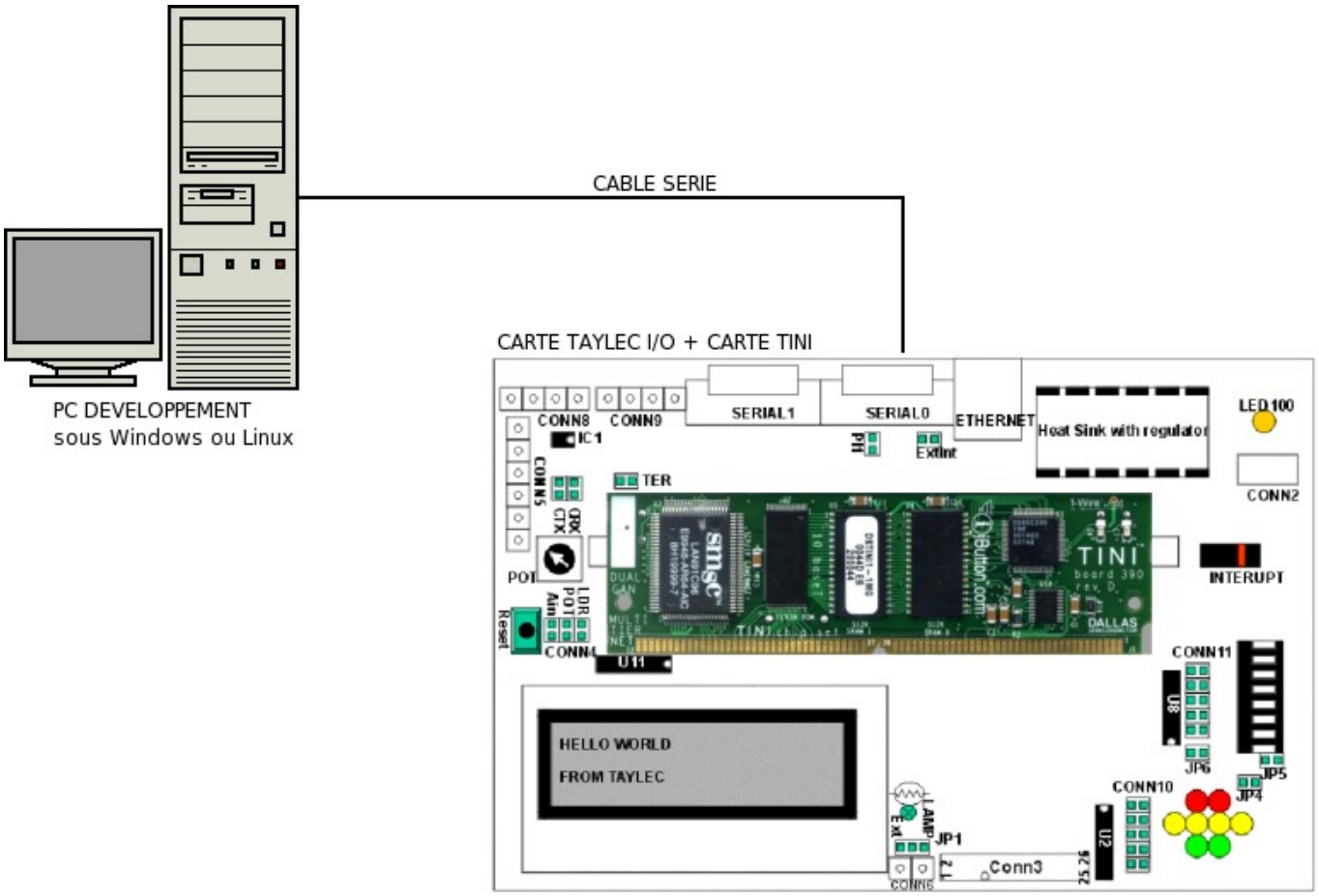
Table des matières

Présentation....2
Introduction....3
Matériel nécessaire....3
Logiciel nécessaire....3
Partie A : Installation du matériel et des ressources logicielles....5
Partie B : Déploiement de l'environnement TINI....9
Partie C : Configuration de l'environnement TINI....12
Partie C : Tests de mise en oeuvre de la carte TINI....18

Remarque : il est fortement conseillé de lire les annexes sur les parties Réseau et Programmation.

PRÉSENTATION

La carte « Tini » étant livrée sans l'OS (TiniOS), il est nécessaire de l'installer et de le configurer avant utilisation.



La liaison série (SERIAL 0) peut servir dans deux situations différentes :

- *en phase de configuration ou de test :* elle peut servir à dialoguer avec la carte Tini en utilisant Slush, l'interprèteur de commandes de celle-ci. Cette fonctionnalité permet, par exemple, de configurer l'adresse IP de la carte Tini. Côté PC, on pourra utiliser pour dialoguer par exemple l'Hyperterminal de Windows ou Minicom sous Linux. L'accès nécessitant une authentification (un compte identifié par un login et un mot de passe), on utilisera l'identifiant suivant : login = root et mot de passe = tini (en respectant la casse)

- *en phase d'intallation :* elle est également utilisée afin de transférer le système d'exploitation TiniOS et l'interprèteur de commande Slush dans la carte Tini (celle-ci étant livrée sans système d'exploitation). On peut aussi utiliser cette fonctionnalité pour mettre à jour le système d'exploitation TiniOS ou pour le recharger en cas de problème (le mot de passe administrateur modifié et oublié, impossibilité d'accéder à l'interpréteur de commandes). La société Dallas (le constructeur de la carte Tini) fournit le logiciel JavaKit pour cette procédure.

INTRODUCTION

Ce manuel d'installation de l'environnement pour la carte TINI se décompose en plusieurs parties :

- Partie A : installation du matériel et des ressources logicielles (→ par le technicien IRIS)
- Partie B : déploiement de l'environnement TINI (PC_DEV → TINI)
- Partie C : configuration de l'environnement TINI (PC_DEV → TINI)
- Partie D : tests de mise en oeuvre de la carte TINI (PC_DEV → TINI)

MATÉRIEL NÉCESSAIRE

Le matériel à votre disposition est le suivant :

Carte Tini avec connecteur SIMM (TINI Board Model 390)

Carte taylec (E10 Socket) supportant la carte Tini

Un poste de développement (compatible PC) avec liaison série RS232 et carte réseau.

Le poste de développement permet de créer, fabriquer et déployer des applications pour le système embarqué TINI. Il sert aussi de support pour déployer l'environnement d'exploitation pour la carte TINI et pour le configurer.

LOGICIEL NÉCESSAIRE

La liste des logiciels est la suivante :

Nom	*Description*	*Rôle*
Windows ©	Windows est une gamme de systèmes d'exploitation produite par Microsoft ©, principalement destinés aux ordinateurs compatibles PC. C'est le successeur de MS-DOS. Windows est distribuée sous la licence Microsoft EULA (End User License Agreement). La plupart des ordinateurs vendus ont un système Windows préinstallé par le constructeur sous licence OEM (Original Equipment Manufacturer). Contrairement aux autres logiciels utilisés ici, Windows fait partie des logiciels propriétaires. Lien : http://www.microsoft.com/france/windows/	permet d'exploiter le poste de développement (PC_DEV).
Linux	Linux, ou GNU/Linux, est un système d'exploitation compatible POSIX. Linux est basé sur le noyau Linux, logiciel libre créé en 1991 par Linus Torvalds pour ordinateur compatible PC. La mascotte de Linux est le manchot Tux. La principale originalité de GNU/Linux par rapport à d'autres systèmes d'exploitation concurrents (comme Mac OS, Microsoft Windows et Solaris) est d'être constitué d'un noyau libre et de logiciels libres. La licence GNU GPL (écrite par Richard Stallman et utilisée par le noyau Linux) accordent quatre libertés : utiliser le logiciel sans restriction, étudier le logiciel, le modifier pour l'adapter à ses besoins et le redistribuer sous certaines conditions précises.	permet d'exploiter le poste de développement (PC_DEV).

Nom	*Description*	*Rôle*
	Les logiciels libres sont produits de manière collaborative, souvent indépendamment les uns des autres, et peuvent être librement redistribués. Il s'ensuit une particularité du monde GNU/Linux : la séparation fréquente entre ceux qui produisent les logiciels et ceux qui les distribuent. On appelle distribution Linux une solution prête à être installée par l'utilisateur final comprenant un noyau Linux, des programmes d'installation et d'administration de l'ordinateur, un mécanisme facilitant l'installation et la mise à jour des logiciels (comme RPM ou APT) ainsi qu'une sélection de logiciels produits par d'autres.	
Java	Java est une technologie développée par Sun Microsystems. Elle correspond à plusieurs produits et spécifications de logiciels qui, ensemble, constituent un système pour développer et déployer des applications. Java est utilisée dans une grande variété de plates-formes depuis les systèmes embarqués et les téléphones mobiles jusqu'aux serveurs, les applications d'entreprise, les superordinateurs et dans une moindre mesure pour les interfaces graphiques comme les applets Java du Web. Java est sous licence GNU GPL depuis novembre 2006. Depuis des années, Sun Microsystems appelle Java la « technologie Java » dans son ensemble. En pratique, beaucoup de programmeurs utilisent le mot « Java » pour désigner le langage de programmation, tandis que la plate-forme d'exécution est appelée « JRE » Java Runtime Environment, environnement d'exécution Java) et le système de compilation : « JDK » (Java Development Kit) plutôt que « compilateur Java ». Lien : http://java.sun.com/	La technologie JAVA est utilisée de la manière suivante : « java » : le langage de programmation de l'application s'exécutant sur la carte TINI « jre » : machine virtuelle présente sur la carte TINI permettant d'exécuter l'application « jdk » : environnement de développement installé sur PC_DEV permettant de fabriquer l'application pour la carte TINI. La version du JDK utilisée dans ce projet est la 1.4.2_19.
HyperTerminal	HyperTerminal est un programme d'établir une connexion entre ordinateurs par le réseau téléphonique (et un modem) ou câble sur le port série (COMx). Intégré à Windows à partir de la version 2.0 (avant Windows 95), c'est une interface graphique légère capable qui se présente en fait comme un bloc-note doté de fonctionnalités de communication.	permet de communiquer par le port série RS232 sous Windows
Minicom	Minicom est un programme de communication qui permet le contrôle de modem et l'émulation de terminal pour les Unix-like, écrit par Miquel van Smoorenburg. Il est distribué sous licence GNU GPL.	permet de communiquer par le port série RS232 sous Linux
Javakit	Logiciel dédié fourni par Dallas Semiconductor et nécessitant un machine viruelle sur la machine hôte.	permet de transférer le système d'exploitation TiniOS et l'interprèteur de commande Slush dans la carte Tini
COMM API	L'API Java Communications (également connu sous le nom de javax.comm) est une extension Java qui facilite le développement indépendant sur des plate-formes de communications pour des applications dédiées aux systèmes embarqués.	permet l'accès au port série RS232 à un programme Java

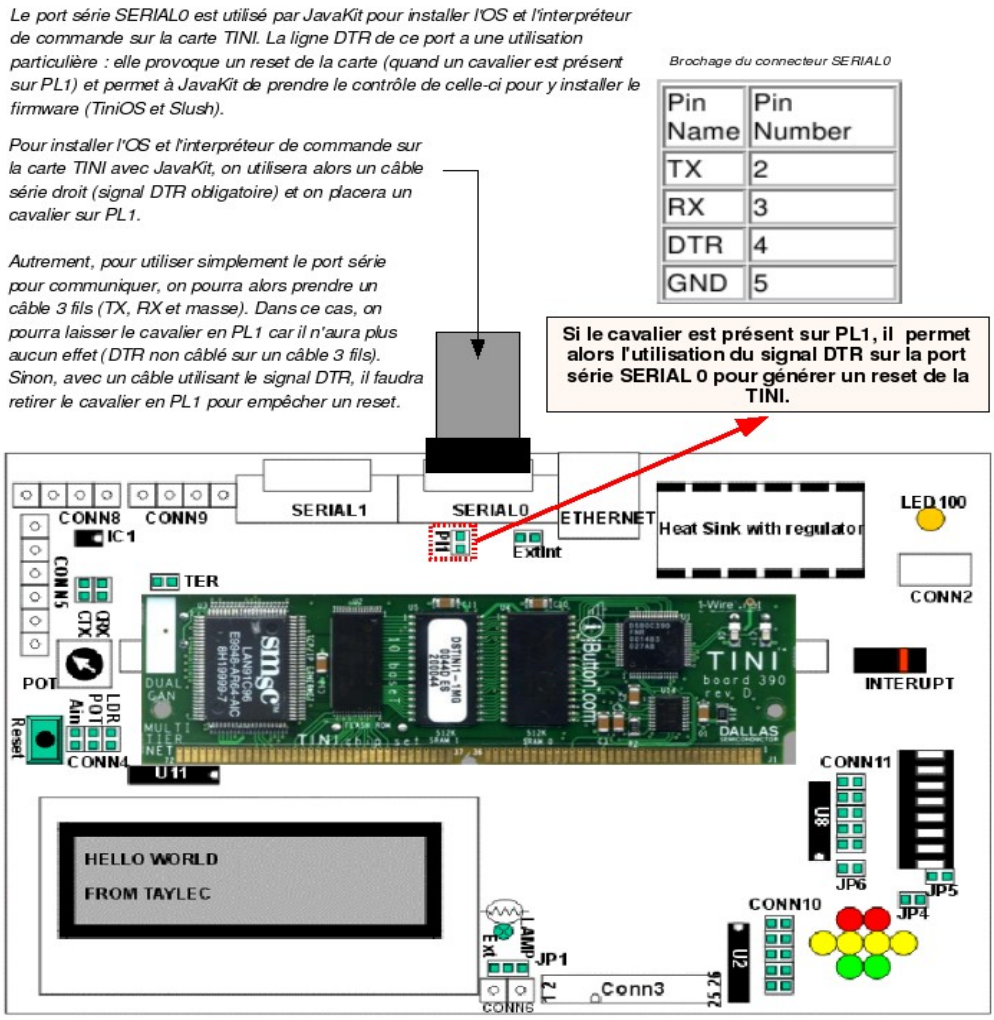
Nom	*Description*	*Rôle*
	Les implémentations de l'API sont actuellement disponibles pour Solaris SPARC, Solaris x86 et Linux x86. Lien : http://java.sun.com/products/javacomm/	
RXTX	RXTX est une bibliothèque Java permettant la communication série et parallèle pour le kit de développement Java (JDK). Tous les produits sont sous la licence GNU LGPL. RXTX est basé sur les spécifications de Sun Java Communications API. Lien : http://www.rxtx.org/	permet l'accès au port série RS232 à un programme Java sous Linux
Notepad++	Notepad++ est un éditeur de texte générique écrit en C++, qui intègre de nombreuses fonctionnalités dont la coloration syntaxique de code source pour beaucoup de langages (C, C++, Java, etc ..). Il est également une alternative au bloc-notes de Windows (d'où le nom). Le projet est sous licence GPL. Lien : http://notepad-plus.sourceforge.net/fr/site.htm	permet d'éditer les fichiers sources d'une application sur le poste de développement (PC_DEV sous Windows) ou les fichiers de configuration du système TINI.
vim	Vim est un éditeur de texte, c'est-à-dire un logiciel permettant la manipulation de fichiers texte. Il est directement inspiré de vi (un éditeur très répandu sur les systèmes d'exploitation de type UNIX), dont il est le clone le plus populaire. Son nom signifie d'ailleurs Vi IMproved, que l'on peut traduire par « VI aMélioré », ou éventuellement « VI Meilleur », ou encore « VI en Mieux ». Contrairement à vi, Vim est un logiciel libre. Vim possède son propre langage d'extension, ce qui lui permet la prise en charge de tâches évoluées, telles que la production de code source. Malgré de nombreuses fonctionnalités, il conserve un temps de démarrage court et reste ainsi adapté pour des modifications simples et ponctuelles (fichiers de configuration par exemple). Lien : http://www.vim.org/	permet d'éditer les fichiers sources d'une application sur le poste de développement (PC_DEV sous Linux) ou les fichiers de configuration du système TINI.

PARTIE A : INSTALLATION DU MATÉRIEL ET DES RESSOURCES LOGICIELLES

A . 1 . Installer le matériel fourni

Vous devez réaliser dans l'ordre les actions suivantes :

- relier la carte Taylec/TINI au PC_DEV par un câble série (voir schéma)
- alimenter la carte Taylec/TINI (adaptateur 12V fourni)
- démarrer le poste de développement PC_DEV sous Windows ou sous Linux



A . 2 . Installer les ressources logicielles.

Avant de passer aux phases d'intsallation et de configuration, il est nécessaire de préparer logiciellement le poste PC_DEV avec les ressources logicielles suivantes :

- un système d'exploitation pour le poste PC_DEV (Windows ou Linux)
- un éditeur de texte (par exemple Notepad++ ou vim)
- le kit de développement Java J2SE SDK
- les APIs de communication (COMM API et RXTX)
- l'environnement TINI

Vérifier préalablement leur présence sur le poste de développement et les installer si nécessaire (ils sont disponibles sur le serveur de la section, sur le CD ressources et par téléchargement sur Internet).

Remarque : ce docuement ne traite pas de l'installation des systèmes d'exploitation Windows ou Linux.

A . 2 . 1 . Installation du JDK

a . Installer la version présente sur le CD ressources (par exemple la version 1.4.2_19 ou en télécharger une plus récente du J2SE SDK). Ce fichier d'installation a un nom sous la forme « j2sdk-VERSION-OS.ext », où VERSION représente une suite de chiffres séparés par des points (ex. : 1.4.1) et OS le système d'exploitation destination du SDK (ex. : windows-i586). Par défaut, l'extension (.ext) sera .exe pour Windows et .bin pour Linux.

Pour Windows :

```
Exécuter le fichier d'installation j2sdkVERSION-OS.exe et installer le SDK dans le répertoire
C:\j2sdkVERSION (par exemple, C:\j2sdk1.4.2_19).
```

Sous Linux :

```
Ouvrir un terminal
Se déplacer dans le répertoire où se trouve le fichier j2sdk-VERSION-OS.bin avec la commande
cd
Ajouter le droit d'exécution au fichier j2sdk-VERSION-OS.bin avec la commande : chmod +x
j2sdk-VERSION-OS.bin
Exécuter le fichier d'installation : ./j2sdk-VERSION-OS.bin
```

b . Ajouter au PATH du système d'exploitation le chemin d'accès au sous-dossier « bin » du SDK contenant les commandes JAVA comme cela est indiqué ci-après :

Sous Windows 95/98/ME :

```
Editer le fichier C:\AUTOEXEC.BAT et ajouter la ligne :

PATH=%PATH% ;C:\j2sdkVERSION\bin

Redémarrer l'ordinateur.
```

Sous Windows 2000/XP :

```
Cliquer sur le bouton droit de la souris sur l'icône de votre poste de travail et choisir le
menu « Propriétés »
Choisir l'onglet « Avancé »
Cliquer sur le bouton « Variables d'environnement ' »
Ajouter la variable d'environnement PATH avec la valeur :
     %PATH%;C:\j2sdkVERSION\bin
Si la variable existe déjà, la modifier en ajoutant à la fin de sa valeur :
     ;C:\j2sdkVERSION\bin
Confirmer en cliquant sur « Modifier »
```

Sous Linux :

```
Editer le fichier ~/.bashrc et ajouter les lignes :
     PATH=$PATH:/chemin/vers/j2sdkVERSION/bin
     export PATH
Redémarrer la session.
```

A . 2 . 2 . Installation de Java Communication API

Cette API fournit le code natif nécessaire pour utiliser le port série du PC (nécessaire pour l'utilitaire JavaKit).

Sous Windows :

```
Placer la DLL « win32com.dll » sous le répertoire <jdk>\jre\bin où <jdk> fait référence à la
racine du SDK.
Placer l'archive Java « comm.jar » sous le répertoire <jdk>\jre\lib\ext
Placer le fichier « javax.comm.properties » sous le répertoire <jdk>\jre\lib.
Remarque : il est inutile de toucher au « CLASSPATH ».
```

Sous Linux :

```
Pour COMM API :
Se placer dans le répertoire racine du JDK : cd <jdk> (où <jdk> fait référence à la racine du
SDK)
Décompresser l'archive javacomm20-x86.tar.Z : tar zxvf javacomm20-x86.tar.Z
Placer le fichier comm.jar dans <jdk>/jre/lib/ext/ : cp commapi/comm.jar <jdk>/jre/lib/ext/
Placer le fichier javax.comm.properties dans <jdk>/jre/lib/ : cp
commapi/javax.comm.properties <jdk>/jre/lib/
Ajouter l'accès au fichier comm.jar à la variable CLASSPATH :
export CLASSPATH=$CLASSPATH:<jdk>/jre/lib/ext/comm.jar

Pour RXTX :
Se placer dans /usr/local/ : cd /usr/local
Décompresser l'archive rxtx-2.0-5.tar.gz dans /usr/local/ : tar zxvf rxtx-2.0-5.tar.gz
Se placer dans le répertoire d'installation : cd /usr/local/rxtx-2.0-5
Lancer la procédure de configuration : ./configure
Fabriquer : make
Installer : make install
```

A . 2 . 3 . Autres ressources

Hyperterminal : si ce programme n'a pas été installé sur votre poste lors de l'installation de Windows, aller dans le panneau de configuration, ajout de programmes, ajout de composants Windows. Vous aurez besoin du CDROM de Windows.

Minicom : si ce programme n'a pas été installé sur votre poste lors de l'installation de Linux, lancer un terminal sous root et installer le programme avec : urpmi minicom. Vous pouvez aussi utiliser l'interface graphique via « Installer des logciels ». Vous aurez besoin du CDROM/DVD de Linux ou d'une connexion Internet.

Notepad++ : utiliser l'installateur fourni

vim : si ce programme n'a pas été installé sur votre poste lors de l'installation de Linux, lancer un terminal sous root et installer le programme avec : urpmi vim. Vous pouvez aussi utiliser l'interface graphique via « Installer des logciels ». Vous aurez besoin du CDROM/DVD de Linux ou d'une connexion Internet.

L'environnement TINI : il est fourni sous la forme d'une archive compressée tiniVERSION.ext. Il est téléchargeable à l'adresse http://files.dalsemi.com/tini/index.html (TINI Firmware Download) et contient (SDK Contents for 1.0 and 1.1) :

- les binaires pour l'OS (firmware) et l'interpréteur de commandes Slush
- les outils pour convertir des applications pour la TINI
- les outils pour charger des applications sur le système TINI
- la documentation de l'API TINI
- des exemples de code source et de scripts
- les outils et la documentation pour fabriquer des librairies natives
- les codes sources des outils, de l'API TINI et de Slush

Sous Windows :

```
Décompresser l'archive tini1_02f.tgz dans C:\
```

Sous Linux :

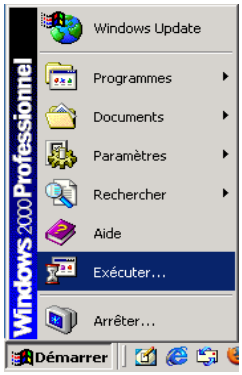
```
Décompresser l'archive tini1_02f.tgz dans /usr/loca/ :
cd /usr/local
tar zxvf tini1_02f.tgz
```

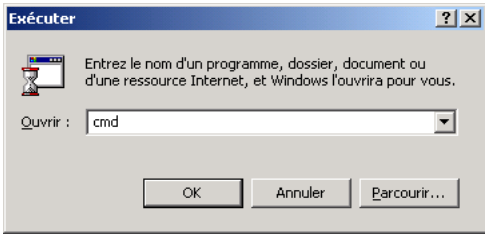
PARTIE B : DÉPLOIEMENT DE L'ENVIRONNEMENT TINI

B . 1 . Exécuter JavaKit.

Cet utilitaire va permettre de transférer le système d'exploitation TiniOS et l'interprèteur de commande Slush sur la carte Tini. Il utilise le port série SERIAL0 pour communiquer avec le système embarqué. JavaKit possède une interface graphique et se lance, depuis l'interpéteur de commande, de la manière suivante :

Sous Windows :



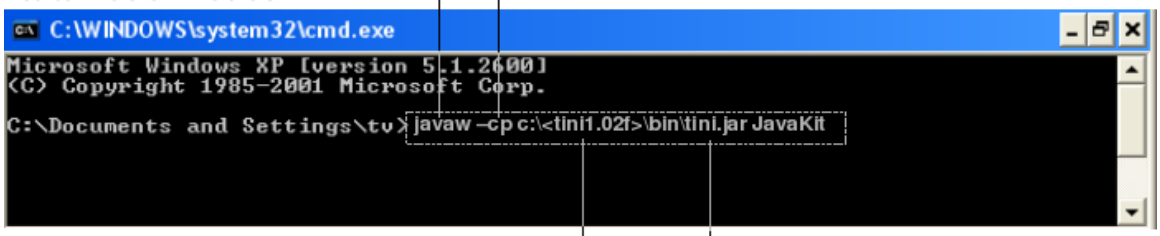


```
javaw -cp c:\tini1.02f\bin\tini.jar JavaKit
```

javaw : cet outil (comme la commande java) est l'interpréteur de bytecode. Il lance le JRE (la machine virtuelle Java), charge les classes nécessaires et exécute la méthode main de la classe.

Remarque : java ouvre une console pour recevoir les messages de l'application alors que javaw n'en n'ouvre pas. La variable d'environnement PATH doit contenir le chemin vers le JDK.

-cp : permet d'indiquer les chemins de recherche des classes nécessaires à l'exécution. Chaque répertoire doit être séparé avec un point virgule. Cette option utilisée annule l'utilisation de la variable système CLASSPATH



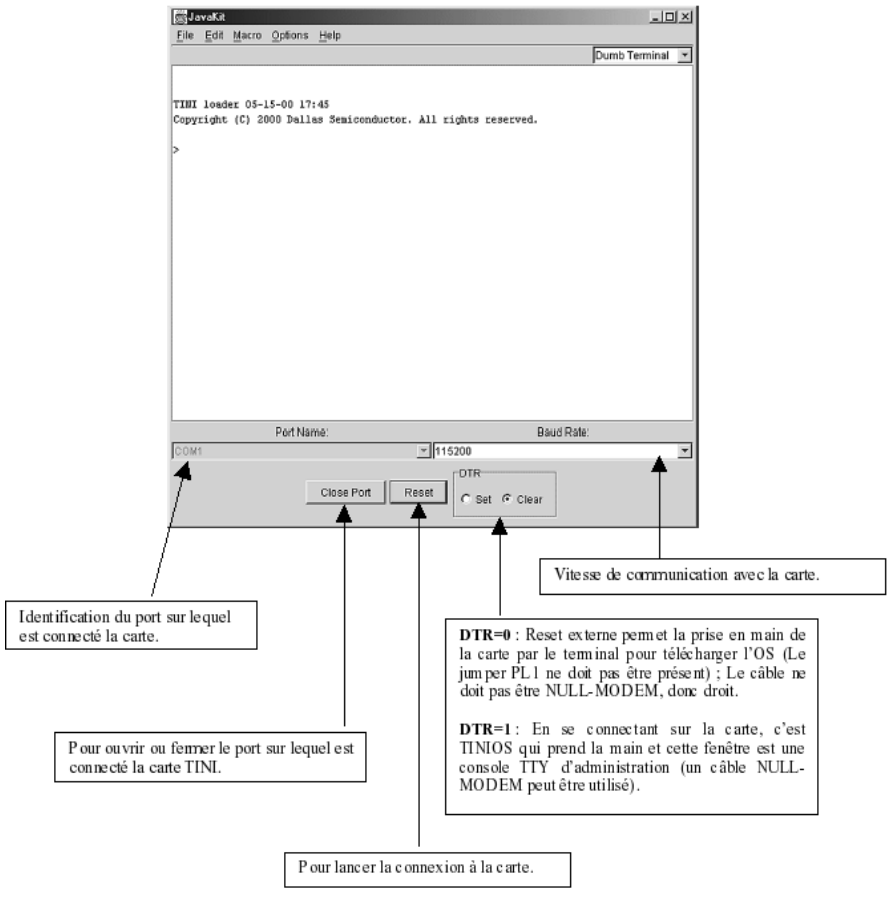
Répertoire où se trouve l'archive tini.jar

JAR est le diminutif de Java ARchive. C'est un format de fichier qui permet de regrouper des fichiers contenant du byte-code java (fichier .class) ou des données utilisées en temps que ressources (images, son, ...). Ce format est compatible avec le format ZIP : les fichiers contenus dans un jar sont compressés de façon indépendante du système d'exploitation.

L'archive tini.jar contient le fichier (ByteCode) JavaKit.

Sous Linux :

```
java -classpath /usr/local/tini1.02f/bin/tini.jar:$CLASSPATH JavaKit
```



B . 2 . Déployer l'environnement TINI avec JavaKit.

Attention : cette manoeuvre entraînera la destruction de tous les fichiers présents sur la carte

Sous Windows ou Linux :

Selectionner le port pour se connecter à la carte (par exemple COM1 pour Windows ou /dev/ttyS0 pour Linux) et choisir la vitesse de transmission (part défaut prendre 115200 bauds)
Cliquer sur Open Port puis Reset.
Après appui sur Reset, le message suivant et un prompt « > » apparaîssent :

```
TINI Loader 05-15-00 17:45

Copyright (C) 2000 Dallas Semiconductor. All rights reserved

>
```

JavaKit est maintenant connecté à la carte.

Aller dans le menu "File" et sélectionner "Load File".
Choisir les fichiers tini.tbin et slush.tbin.
Appuyer sur le bouton « Open », les messages suivants doivent apparaître :

```
Loading file: C:\tini1.02\bin\slush.tbin.

Please wait... (ESC to abort.)

Load complete.



Loading file: C:\tini1.02\bin\tini.tbin.

Please wait... (ESC to abort.)
```

Le chargement va prendre quelques secondes.
Après le transfert, la carte est sous le contrôle du loader. vous pouvez lister les commandes en tapant :

```
> ? <Entrée>
```

Ensuite, on réinitialise la carte en sélectionnant la banque 18 de la mémoire « Heap » (premiers 64 KO de la mémoire) puis l'initialiser à 0 :

```
> b18 <Entrée> // permet de se placer sur la bank 18
> f0 <Entrée> // remplie (fill) la bank 18 avec des 0 (donc l'efface)
```

La carte est maintenant prête à « booter » avec l'OS ; pour ce faire, taper E (exit) pour redonner la main au « BootLoader ». A ce stade, deux comptes d'identication sont disponibles : administrateur (root/tini) et invité (guest/guest).

L'étape suivante est la configuration du système TINI. Elle peut se réaliser à partir de JavaKit ou d'un émulateur de terminal série comme l'HyperTerminal (pour Windows) ou Minicom (pour Linux).

Pour quitter JavaKit, il suffit de fermer le port « Close Port » et fermer la fenêtre.

PARTIE C : CONFIGURATION DE L'ENVIRONNEMENT TINI

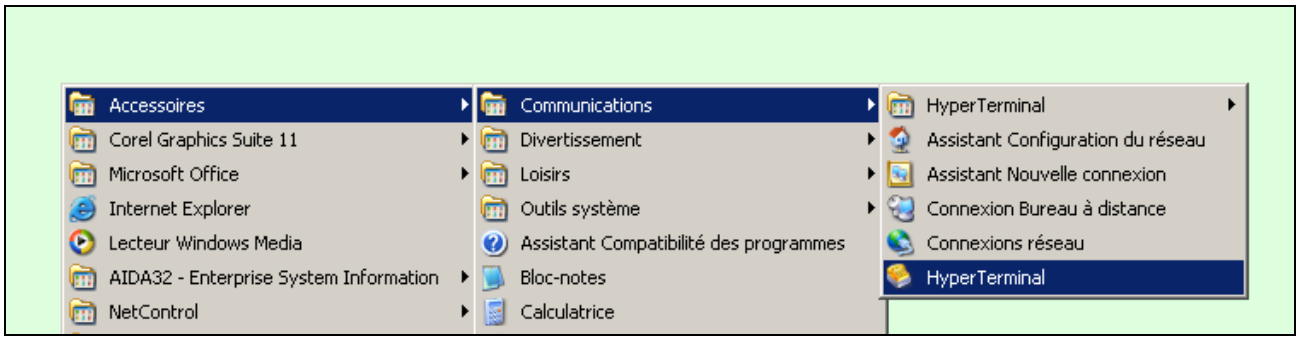
On suppose que l'accès réseau n'est pas encore configuré. Cette étape de configuration nécessite donc un accès par le port série (SERIAL0) pour être réaliser.

On peut donc soit utiliser JavaKit ou un émulateur de terminal série comme l'HyperTerminal (pour Windows) ou Minicom (pour Linux).

C . 1 . Utilisation de l'HyperTerminal

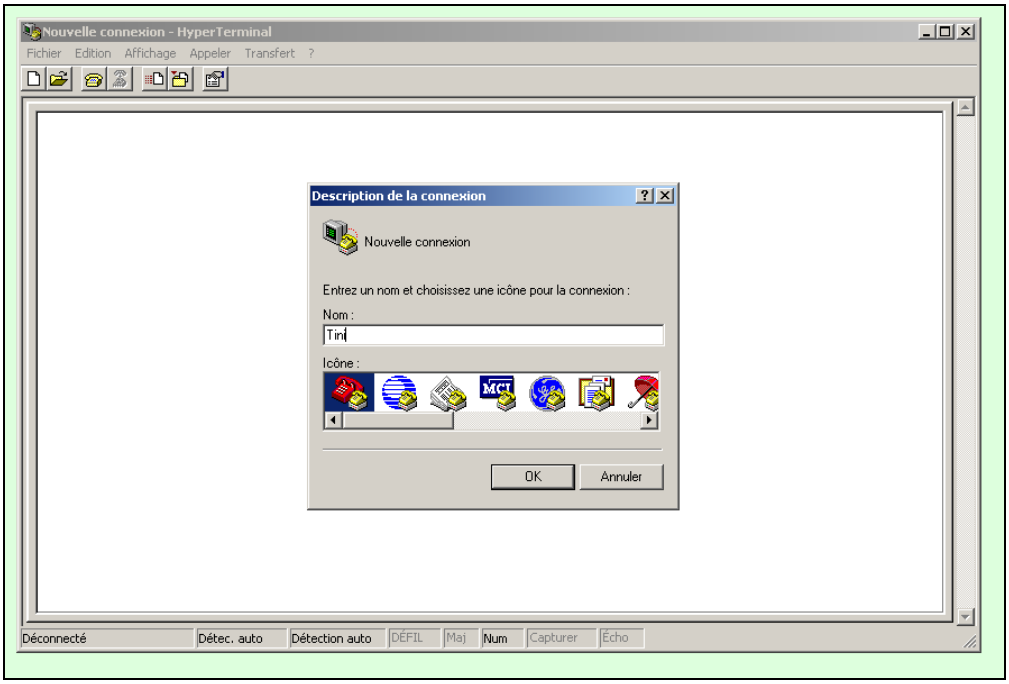
On va utiliser l'HyperTerminal de Windows, pour cela il faut faire :

```
Menu Démarrer → Accessoires → Communications → HyperTerminal
```



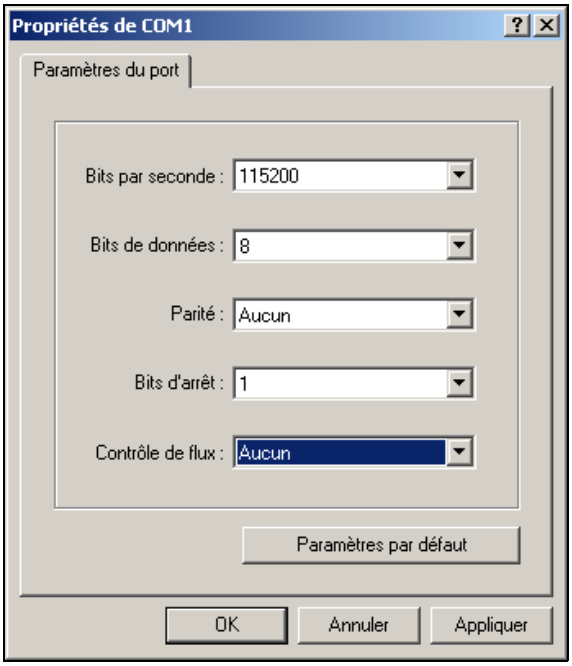
Remarque : Si ce programme n'a pas été installé sur votre poste lors de l'installation de Windows, aller dans le panneau de configuration, ajout de programmes, ajout de composants Windows. Vous aurez besoin du CDROM de Windows.

On va maintenant définir un profil de session :

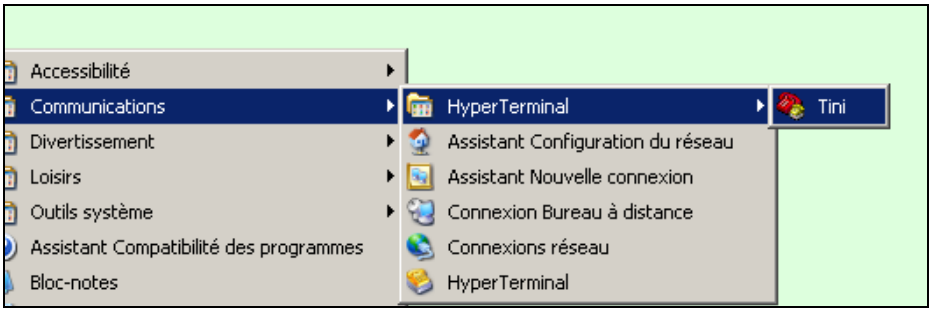


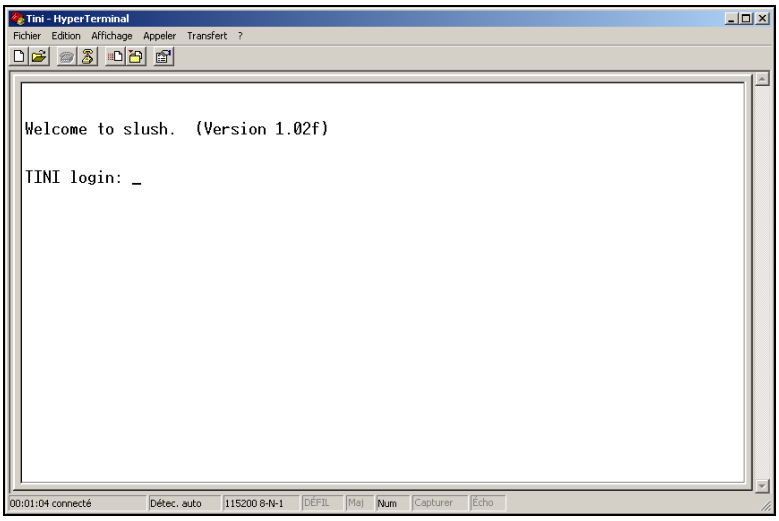
Il faut paramétrer les propriétés du port COMx (115200 bits par secondes, 8 bits de données, pas de parité, un bit d'arrêt, pas de contrôle de flux).

On termine en enregistrant la session (fichier, enregistrer sous 'Tini.ht').



On peut maintenant démarrer la session Tini :





Pour quitter, on fermera simplement la fenêtre de l'HyperTerminal.

C . 2 . Configuration du réseau

La re-mise sous tension ou reset de la carte Tini permet d'observer ce qui se passe au démarrage (*boot*) de la carte Tini :

```
----> TINI Boot <----

TINI OS 1.02f
API Version 800E
Copyright (C) 1999 - 2003 Dallas Semiconductor Corporation
E9000000
Running POR Code
Memory POR Routines
000028
0080,0100,0180,0200,0280,0300,0380,0400,0480,0500,0580,0600,0680,0700,0780,0800,
0880,0900,0980,0A00,0A80,Transient blocks freed: 0A95, size: 034660
CPersistant blocks freed: 0000, size: 000000
KM_Init Passed

TTS Revision: 188 , Date: 1/21/02 8:15a
Thread_Init Passed
External Serial Port Init
External serial ports not enabled
Memory Available: 0E5420
Creating Task:
0100
01
Loading application at 0x070100
Creating Task:
0200
02
Application load complete


[-=        slush Version 1.02f        =-]
[          System coming up.            ]
[      Beginning initialization...      ]
[        Not generating log file.       ]    [Info]
[    Initializing shell commands...     ]    [Done]
[        Checking system files...       ]    [Done]
[ Initializing and parsing .startup...  ]
[        Initializing network...        ]
[     Starting up Telnet server...      ]    [Done]
[       Starting up FTP server...       ]    [Done]
[         Network configuration         ]    [Done]
[         System init routines          ]    [Done]

[    slush initialization complete.     ]

Hit any key to login.
```

Après appui sur une touche,

```
Welcome to slush.  (Version 1.02f)
```

```
TINI login:
```

On doit s'identifier pour pouvoir communiquer avec la carte embarquée :

```
TINI login: root
TINI password: tini


TINI />
```

Remarque : le mot de passe tini est masqué lors de la saisie

On utilisera la commande **ipconfig** :

- pour consulter les paramètres réseau de la carte tini

```
TINI /> ipconfig
Hostname          : TINI.
Current IP        :
Default Gateway   :
Subnet Mask       :
Ethernet Address  : 00:60:35:00:a0:19
Primary DNS       :
Secondary DNS     :
DNS Timeout       : 0 (ms)
DHCP Server       :
DHCP Enabled      : false
Mailhost          :
Restore From Flash: Not Committed
```

- pour configurer les paramètres réseau de la carte tini

```
TINI /> ipconfig -a 192.168.52.254 -m 255.255.255.0
Warning:  This will disconnect any connected network users and reset all network servers.
OK to proceed? (Y/N): y
[ Tue Feb 21 13:01:32 GMT 2006 ]  Message from System: FTP server stopped.
[ Tue Feb 21 13:01:33 GMT 2006 ]  Message from System: Telnet server stopped.
[ Tue Feb 21 13:01:33 GMT 2006 ]  Message from System: Telnet server started.
[ Tue Feb 21 13:01:34 GMT 2006 ]  Message from System: FTP server started.


TINI />
```

Explications :

Hostname	Nom d'hôte du système (résolu par le serveur DNS)
Current IP	Adresse IP de la carte
Default Gateway	Adresse IP servant de passerelle pour accéder à un autre sous-réseau.
Subnet Mask	Masque de sous réseau
Ethernet Address	Adresse unique MAC de la carte
Primary DNS	Adresse IP du serveur DNS
Secondary DNS	
DNS Timeout	
DHCP Server	Adresse IP du serveur dynamiqued'@ IP.
DHCP Enabled	permet d'activer le service d'adressage IP dynamique au démarrage
Mailhost	Adresse IP du poste servant de serveur de messagerie
Restore From Flash	

C . 3 . Configuration de la TINI

Les fichiers de configuration de l'environnement TINI sont situés dans le répertoire /**etc** et sont au nombre de trois :

- **passwd** : qui définit les comptes d'identification (login et mots de passe crypté avec l'algorithme Secure Hash Algorithm « SHA1 ») présents sur le système

```
root:f8491b67e91f837c13c3444965281bcee5fca964:128

guest:1bb6e3a2abc20f654fe62fd139790c06394885d3:0
```

- **.tininet** : qui définit le nom d'hôte et de domaine du système

```
HostName:TINI

DomainName:
```

- **.startup** : est interprété par le « Slush » à chaque démarrqge (*boot*). Ce fichier permet de configurer les variables d'environnement des services (FTP, Telnet et série) et de lancer automatiquement une application

```
#Autogen'd slush startup file

setenv FTPServer enable

setenv TelnetServer enable

setenv SerialServer enable

initializeNetwork
```

Explications :

Les lignes précédés d'un « # » sont des commentaires.
Les trois lignes commençant par « setenv » valident (enable) respectivement les serveurs FTP, Telnet et Série.
Si une application a besoin d'utiliser la liaison série SERIAL0 que Slush utilise par défaut comme serveur série, l'administrateur devra mettre cette ligne en commentaire.

Des applications peuvent être chargées à l'interprétation de ce fichier, par une commande appropriée comme, par exemple :

java Blinky.tini > debug.log

Cette commande demande à Slush d'exécuter le programme Blinky.tini et de rediriger les flux java.lang.System.out et java.lang.System.err vers un fichier de journalisation (log) nommé debug.log.

Toutes les applications lancées à partir de ce fichier le seront en tâche de fond.

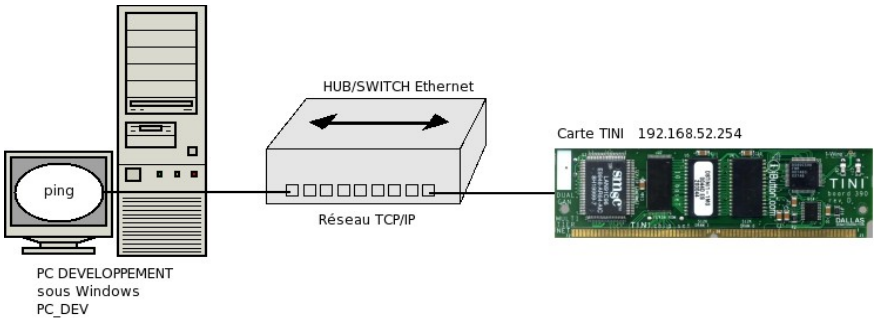
PARTIE C : TESTS DE MISE EN OEUVRE DE LA CARTE TINI

Cette partie est détaillée dans une séquence de travaux pratiques. Pour en savoir, se référer donc à ce document.

Remarque : pour réaliser l'ensemble des tests présentés ci-dessous, vous aurez besoin d'outils réseaux comme un client Telnet (par exemple, PuTTY pour Windows ou la commande telnet) et d'un client FTP (par exemple, FileZilla pour Windows ou la commande ftp).

C . 1 . Tester la liaison réseau avec le système embarqué (carte TINI).

Pour réaliser un test de base d'une liaison réseau, on utilise souvent la commande **ping** qui permettra, en cas de succès, de valider la pile de protocoles jusqu'au niveau IP.



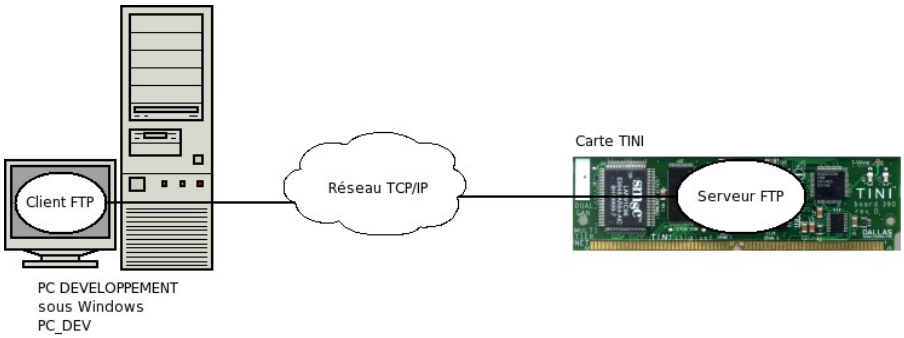
En utilisant l'interpréteur de commandes, taper la commande :

```
ping adresse_ip_tini
```

C . 2 . Tester le déploiement d'une application sur le système embarqué cible.

La carte TINI dispose d'un serveur FTP natif. On va donc utiliser un client FTP (comme FileZilla) pour transférer un programme **.tini** vers la carte TINI.

FTP (*File Transfer Protocol*, protocole de transfert de fichiers) est un protocole de communication dédié à l'échange de fichiers informatique sur un réseau TCP/IP.



On peut utiliser le logiciel FileZilla pour ouvrir une connexion FTP et transférer le programme **.tini** vers la carte TINI.

L'ouverture d'une session FTP sur la carte TINI nécessite une authentification (un compte identifié par un login et un mot de passe). Vous utiliserez l'identifiant suivant :

```
login : root

password : tini
```

Pour se connceter au serveur FTP de la carte TINI, vous pouvez utiliser le bouton **Connexion rapide** comme indiquer dans la capture d'écran fournie.

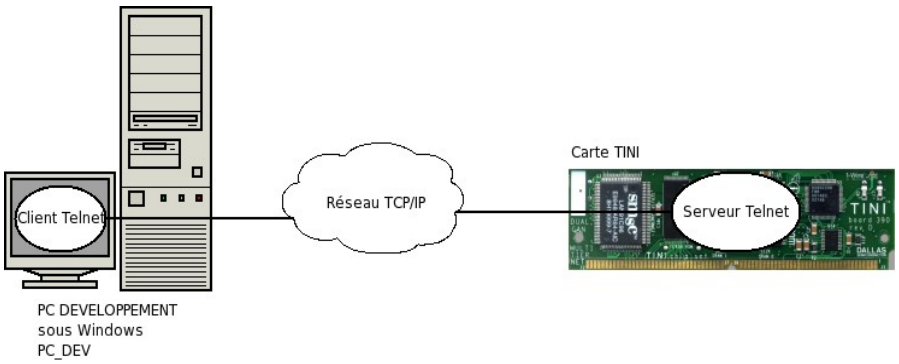
Une fois connecté, le transfert d'un fichier se fait par un simple glisser/déposer avec la souris.

Pour ce test, on utilisera un programme fourni avec l'environnement TINI :

```
<dossier tini>/examples/Blinky/bin/Blinky.tini
```

C . 3 . Tester l'exécution d'une application sur le système embarqué cible.

La carte embarquée TINI ne disposant pas de terminal opérateur (clavier/écran), on va donc utiliser celui du poste de développement par l'intermédiaire d'un logiciel d'émulation de terminal via le réseau. Le protocole permettant l'émulation de terminal en réseau se nomme **telnet**. Le logiciel Putty va nous permettre d'ouvrir une session (accès à un terminal de connexion via le réseau sur la carte TINI).



L'ouverture d'une session sur la carte TINI nécessite une authentification (un compte identifié par un login et un mot de passe). Vous utiliserez l'identifiant suivant :

```
login : root

password : tini
```

Les paramètres de connexion sont les suivants :

```
Host Name (or IP address) ← indiquer l'adresse IP de la carte TINI

Port ← 23 (le numéro de port du service Telnet)

Connection type : Telnet
```

Avec PuTTY, ces paramètres peuvent être sauvegardés dans un profil de session avec le bouton Save et ré-utiliser plus tard avec le bouton Load. Pour ouvrir une session, il faut ensuite cliquer sur le bouton Open !

Une fois connecté, il est possible d'exécuter des commandes sur la carte TINI. Ceci est possible car celle-ci dispose d'un système d'exploitation (TiniOs) et d'un interpréteur de commandes (nommé SLUSH).

```
ls
```

Dans la session Telnet ouverte sur la carte TINI, lancer l'exécution du programme **.tini** avec la commande **java** (machine virtuelle), par exemple :

```
java Blinky.tini
```

Le programme de test Blinky.tini permet de faire clignoter une Led de la carte TINI.

En cas de problème (par exemple si l'application ne répond plus), il est possible d'arrêter le processus en cours de la manière suivante :

- Ouvrir une deuxième session telnet avec PuTTY
- Lister les processus, par exemple :

```
ps
```

- Tuer le processus désiré, par exemple :

```
kill 3
```